Dealing with a medical dataset

Aymane HAOULANI, Antoine TURKIE, Redouane HADNI
Ecole Polytechnique Féderale de Lausanne, Switzerland

I. Preprocessing

A. Data Cleaning

Most columns in our training dataset have significant missing values. Dropping these columns would lead to substantial data loss. Instead, we kept columns with less than 20% missing values, a trade-off that retains columns likely to hold meaningful information.

B. Missing values imputation and Scaling

1) Imputation and Standardization: A straightforward approach to impute missing values is by replacing them with the column mean (or median). To scale values to a standardized range, we use standardization to bring values to $[0, 1]$. This approach performed well in practice, though it raises questions about rigor for categorical data.

2) Handling Categorical Columns: To handle categorical columns, we applied the following steps:

- Classified columns as numerical or categorical, using a threshold of approximately 50 unique values.
- Imputed missing values in categorical columns using the mode, rather than the mean.
- Limited categories to a maximum of six levels per column (five most common levels + "other").
- Applied frequency encoding to map categorical values to $[0, 1]$.

We experimented with various encoding methods. However, most approaches either suffered from data leakage, were heavily influenced by class proportions (e.g., SMOTE), or led to excessive column expansion (e.g., one-hot encoding). This procedure offered a balanced solution for handling categorical data efficiently.

3) Conclusion: While our categorical handling approach seemed well suited to the task, its performance was slightly worse than the straightforward method (F1-score of 0.40 vs. 0.42). One possible reason is frequency encoding: though it avoids issues like data leakage and imbalance, it can cause levels with similar proportions to appear indistinguishable to the algorithm.

C. Data Balancing

By inspecting the true labels present in y_{train}, it is clear that our training dataset is highly imbalanced (Proportion of 1s: 8.83% , Proportion of -1s: 91.16%).

While methods like logistic regression are more robust to imbalanced data compared to other simplistic/straightforward methods like linear regression, most machine learning techniques remain somewhat sensitive to highly imbalanced data.

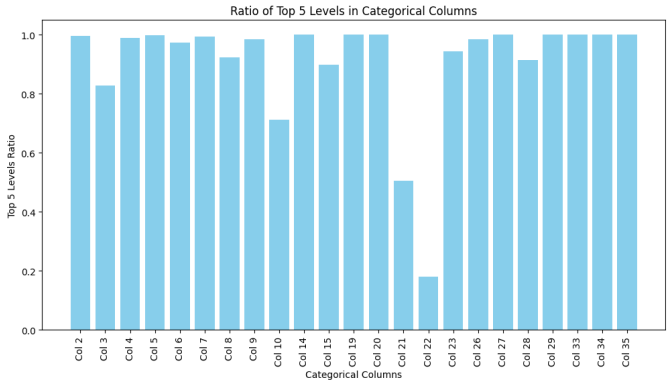

Fig. 1. Why does it make sense to use 6 levels? (21 and 22 were considered numerical afterwards)

In order to tackle this issue, we started by inspecting simple undersampling and oversampling methods. The first method suffers from important loss of data (around 83%) which leads to underfitting, whereas the second one highly overfits to the minority class.

To further improve our model's performance and address class imbalance, we explored advanced ensemble-based balancing techniques. The core approach involves creating multiple balanced subsets from the initial training dataset, then training separate models on each subset. For final predictions, we aggregate the outputs by averaging the computed probabilities across models and determining an optimal decision threshold.

In this context, we explored two main methods:

1) Building 9 balanced datasets with no overlap by duplicating the minority class: This method involves constructing nine distinct balanced datasets, each containing the entire minority class but different, non-overlapping subsets of the majority class. This approach allows us to leverage the full information content of the original training data without redundancy or duplication among majority class samples.

Unlike traditional oversampling, where the same minority data points are duplicated within a single dataset, leading to repeated exposure and a higher penalty for misclassification, our approach distributes the representation of these points across separate datasets and models. This setup minimizes the risk of excessive weighting associated with any single minority class data point. Instead, each model independently evaluates slightly different distributions of the data. Consequently, at prediction time, the final decision is derived by averaging the probabilities across models, which is conceptually similar to

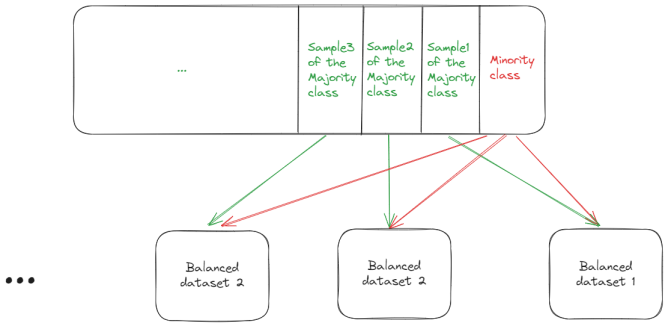

Fig. 2. Building multiple balanced datasets by duplicating the minority class

each minority data point being encountered only once by the ensemble.

This is all great, but we are still learning minority class patterns from only 8.83% of our training data. Can we do better?

2) Building 7 balanced datasets with no overlap by synthetization: In order to ensure that our trained model generalizes well to a wide variety of testing sets, we propose to synthetise new minority class data points using the following procedure:

- Compute the centroids of the minority and majority classes
- The synthetized point would be a convex combination between a real data point of the minority class and a point close to the boundary :

$$\alpha\, x_{real-minority-point} + \frac{1-\alpha}{2}(centroid_{minority} + centroid_{majority})$$

Each one of our subsets will include : Minority class of size N + Synthetized minority class points of size N/2 + Sample of majority class of size 3N/2. By positioning synthetic points close to the boundary between classes, we are likely to improve the model's ability to discern nuanced differences at this critical decision threshold. This should help the model generalize better when it encounters similar boundary cases in test data.

D. Polynomial Feature Expansion

1) $\phi(x) = (1, x, x^2, x^3, ..., x^d)$*:* Relationships between any given feature and the true target variable are not necessarily linear, so it's crucial to explore other types of dependencies.

Using the Taylor expansion, we know that any sufficiently smooth function can be approximated by a polynomial. Therefore, we computed various powers of each initial feature up to a certain degree, allowing us to capture more complex, non-linear patterns. We then select the optimal degree through a grid search.

2) $\phi(x, y) = (1, x, xy, y, x^2, x^2y, y^2x, y^2, ..., x^d, y^d)$*:* We also aimed to explore polynomial expansions with interaction terms, but the computational cost is prohibitive. Even after removing columns with many NaNs and those least correlated with y, the feature count remains very high: Let n=100 be the number of columns after the filtering.

- Degree=2 : 5151 columns
- Degree=3 : 176 851 columns

II. Prediction

A. Model Selection

Logistic regression, which is generally more effective than standard linear models for classification due to its reduced sensitivity to outliers and improved handling of class imbalance, is trained on each of our balanced datasets. For the final prediction, we apply ensembling by averaging the computed probabilities across all logistic regression models. This approach has proven robust, delivering very promising results.

While we also experimented with the Random Forest algorithm, its computational cost was prohibitively high, even with a small number of trees.

B. Parameters Fine-tuning

To maximize the performance of our model, we implemented several procedures:

- Introduced a bias term to better control the decision region.
- Reused the optimal weights from previous models to accelerate convergence, reducing the maximum number of iterations for our gradient descent.
- Determined the optimal degree for polynomial expansion and the best threshold for decision-making using grid search.

C. Results

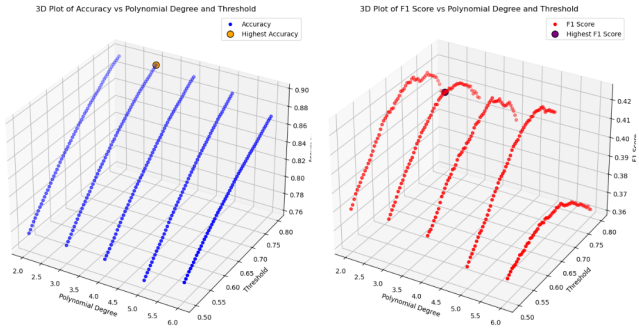

Fig. 3. Parameters fine-tuning for the duplication approach

Both approaches (balancing with duplication and balancing with synthetization) showed very similar performances on the training and testing data.

Nevertheless, we expect the synthetic approach to have more generalization power for other datapoints.

Approach	Optimal Degree	F1 Score (Test)	Accuracy (Test)	F1 Score (Train)	Accuracy (Train)
Synthetization	3	0.421	0.884	0.422	0.864
Duplication	3	0.429	0.864	0.423	0.859

TABLE I
PERFORMANCE COMPARISON OF BALANCING APPROACHES